QLUB

News for the QL user

ISSUE No. 6 JULY/AUGUST 1985

QL, software win awards

The QL and the four business software packages included with it – QL Quill, Abacus, Archive and Easel – won awards for excellence at the British Microcomputing Awards last month.

The prestigious awards, sponsored by Thames Television, The Sunday Times and VNU Business Publications (publishers of Personal Computer World and What Micro?), were given to representatives of Sinclair and Psion (who wrote the software). The QL was given Personal Computer World's award as 'Home Computer of the Year' (beating the Amstrad and MSX computers, which were also nominated in the category), while Psion's QL software earned it the title of 'Home Software of the Year.'

It was the crowning achievement in a year which has seen much activity in connection with both the QL and its popular software. For example, Psion announced late last year that the powerful QL software suite would soon be available in an integrated form for the IBM PC and ACT Apricot computers under the name 'Xchange'. The programming languages and user interface on the QL and IBM/ACT versions are the same, so that programs written in the Archive database language will run on the PC.

The Xchange software earned Psion a nomination for the 'Business Software of the Year award.' The winner in this category had not yet been announced at press time.

Sir Clive Sinclair accepted the 'Personal Computer World Home Computer of the Year' award for the QL at a presentation ceremony in London last month.

INSIDE QLUB NEWS

New QL magazine and peripherals

QLUB News has negotiated a deal which will result in all QLUB members receiving a new QL magazine free for six months.

The new magazine, QL World, is being distributed along with QLUB News for six months and will further expand the information offered to members by QLUB and QLUB News. Your free limited subscription to QL World is the result of a deal negotiated on your behalf by Sinclair with the publishers of QL World.

This means that every issue you'll get up to 60 pages of news, views and how-tos for your QL – in addition to your regular QLUB News. The deal with QL World is part of an on-going effort by Sinclair Research to improve and expand support for the machine.

Every issue of QLUB News will continue to offer a wide breadth of information on or about the QL. Meanwhile, the first issue of QL World is distributed with this issue of QLUB News – and the details of the scheme are explained in the Editor's message on page 2.

* * *

Sinclair Research will shortly be launching a dedicated printer for use with the QL. It will be produced in QL black and offer all the facilities you would expect of a high-quality dot-matrix printer with the additional facility of offering a superior typeface with its Near Letter Quality mode (more details are available on Page 4).

The printer can be used with the four Psion sofware packages which come with the QL so that, for example, business letters produced in QL Quill can be printed using the Near Letter Quality mode with your own headed stationery.

In addition to the printer, Sinclair has almost completed an agreement with Micro Peripherals for the supply of Sinclair-brand 3.5 inch disk drive and disk interface systems for the QL (again details can be found on Page 4).

sinclair

News of the World

By now you're probably wondering why you've got an extra magazine with this issue of QLUB News – and how that happy state of affairs came to be.

Part of this is explained in our news story on Page 1, but the rest will be detailed here. QL World is the latest new magazine devoted exclusively to users of the Sinclair QL and Sinclair have negotiated a deal for QLUB members whereby you can receive the new magazine free for a limited period.

Our negotiations with QL World were partly prompted by our desire to satisfy a demand from you – the QLUB readers – for more and more information about the QL. The new arrangements not only give you QLUB News – with all its regular features – but also a whole new set of features, listings and reviews from QL World.

The QLUB Editor – who was complaining he didn't get enough sleep while having to produce one QLUB News every two months – is also happy with the arrangement as it gives you more information than he currently has room to offer. So we came to the agreement with QL World and obtained six months' worth of free QL World issues for you, our members, and arranged for it to be distributed with QLUB News.

Not only will you continue to receive QLUB News, but you also get a new glossy magazine free with your QLUB News every issue - and the first issue of QL World is enclosed with this issue.

Please write...

You'll note that there is a word-processing package called QL Quill included with your QL. Unbeknown to you, that word-processing package was not put there for you to write reports, business letters and other such trivia — but was solely included for the purpose of writing letters to the Editor of QLUB News.

Said editor was been recently underwhelmed by the volume of mail coming into these offices and has decided to point this little-known fact out to you. He would also like to add that letters written with old-fashioned pen and paper (or even typewriter) are also acceptable.

So, if you have anything to say about QLUB, QLUB News or the QL, please write to him (he's a lonely man) at;

Editor
QLUB News
Sinclair Research
25 Willis Road
Cambridge CB1 2AQ

Q LETTERS

Printer Problems

I am using version 1.01 ADB of Easel which was provided with the purchase of my Sinclair QL. The printer driver provided on this version of QL Easel is for the FX 80 Printer of Epson.

I am using an Epson FX 100 printer and the printer driver does not work. Could you please inform me if there is another printer driver for this programme available and if so where I could get it, or how the existing driver can be modified to suit my printer?
J Stacher

Editor's reply: A larger number of printers are supported in the Version 2 Psion software for QL Easel, Abacus, Archive and Quill. And your membership in QLUB does entitle you to Psion software support – which is probably the best way to get specific technical queries answered.
They can be reached at;
Psion Software Support
22 Dorset Square
London N1

Any Commercial Use?

I have enjoyed using the QL for many business applications, and I think it has great potential. But does it have any great value in the commercial market?

If security protocols – passwords, or encryption facilities – were available, then I think it would have an outlet in firms and offices.

But commercial enterprises will be wary of it so long as strangers are capable of breaking and entering. I would only buy an accounting package for the QL if the data created was safe from outside interference.
N Silver
London EC1

Editor's reply: Many small businesses in particular do not require the high levels of information security demanded by the larger corporations. Since most QL software has been aimed at such users, the software houses have not given such a high priority to security.

Maybe your letter – and letters like it – will inspire them to develop security-oriented financial software.

In addition to the standard software 'lock-outs' the QL's ROM cartridge port could be used as a sort of physical key to sensitive data. And only those with top clearance would have a ROM cartridge to 'unlock' sensitive data. But again, it all depends on demand.

WELCOME TO THE QLUB

This is the first newsletter that you, as a new QLUB member, will receive over the next twelve months.

Psion Software Support Limited supply a comprehensive support service on QL Abacus, Archive, Easel & Quill, Qdos, SuperBASIC and any related peripherals – eg. Printers or memory expansion boards. Psion may be contacted by telephone on 01-723 0553 or by writing to:

Psion Software Support
22 Dorset Square
LONDON NW1

SOFTWARE UPDATE

When this page started last year, we were able to give you detailed descriptions of a few new QL software products each issue. Then we had to have a special pull-out Software Update to accommodate all the new releases. And now there are so many new QL software titles that we can select only some of the most interesting each month and bring them to your attention.

This issue there are several types of software – including leisure software and utilities. Here's a sample:

■ You can now count on your QL for help in the season's planting with QL Gardener by Gardian Computing Services. Even the QL Editor – a self-confessed pink thumb – is hoping this program will finally help turn it a little green.

QL Gardener is a plant-matching and selection program, suitable for use by amateurs, professionals and students. The database contains 25000 pieces of information on more than 1100 species. Users can specify up to 22 characteristics ranging from soil to environmental factors – from which to search for their ideal plant.

For example, you could search for a yellow flowering shrub for borders, which grow in an acidic soil. It is also possible to add your own plant data and additional 'plant data files' will be available from the authors. The comprehensive manual was written by professional horticulturalists and the program is compatible with floppy disks. This is quite simply the most powerful program of its kind currently available. The retail price will be £24.95.

■ But if you're determined to stay inside, and want to do some heavy-duty programming, the QL Macro Assembler by GST has to be the flavour of the month for you. It promises to be one of the most substantial program development tools available for the machine and is a must for every serious QL software developer. The program will retail at £59.95 and will be available in the near future.

■ Equally important for programmers is the long-awaited C programming language developed by Lattice and distributed by Metacomco (who are featured in this issue's Qorporate QLose-up). You'll find this version of C has all the power you would expect from a 6800-family implementation and should allow you the room to develop the tools you need. It will sell for £99.95.

■ The highly-structured (and much-favoured) Pascal programming language has also been developed for the QL by Metacomco. To provide you with the maximum space for your data, part of the program is supplied on a ROM cartridge which slots into the back of the QL. Both this program and the C compiler will be available later this summer. Contact Metacomco for details at 26 Portland Square, Bristol, BS2 8RZ.

* * *

Despite the growing number of games for the QL, Sinclair Research is keen to further broaden the choice of quality entertainment software. If you have software you think could be of publishable quality, send it to:

Games Software Editor
QLUB News
25 Willis Road
Cambridge
CB1 2AQ

Software Specials

To order DIRECT FROM SINCLAIR RESEARCH Ltd: By mail: complete the coupon below. By telephone: please phone 0276 685311 9am–5pm, Monday—Friday (remember to have your credit number at hand).
The following titles are available at a 20% discount from this issue of QLUB: QL Monitor (QLUB price £19.95), QL Toolkit (QLUB price £19.95), and QL Assembler (QLUB price £31.95).
Other software immediately available from Sinclair Research includes QL Chess, QL Touch 'n' Go, QL Cash Trader, QL Project Planner, QL Decision Maker and QL Entrepreneur.

Qty	Item	Item Price	Total

£ _____

*Please delete/complete as applicable Please tick if you require a VAT receipt ☐
*I enclose a cheque/postal order payable to Sinclair Research Ltd. for £ _____
*Please charge to my Access/Barclaycard/Trustcard No. | | | | | | | | | | | | | | | | |

Signature _____ PLEASE PRINT

Name: Mr/Mrs/Miss _____

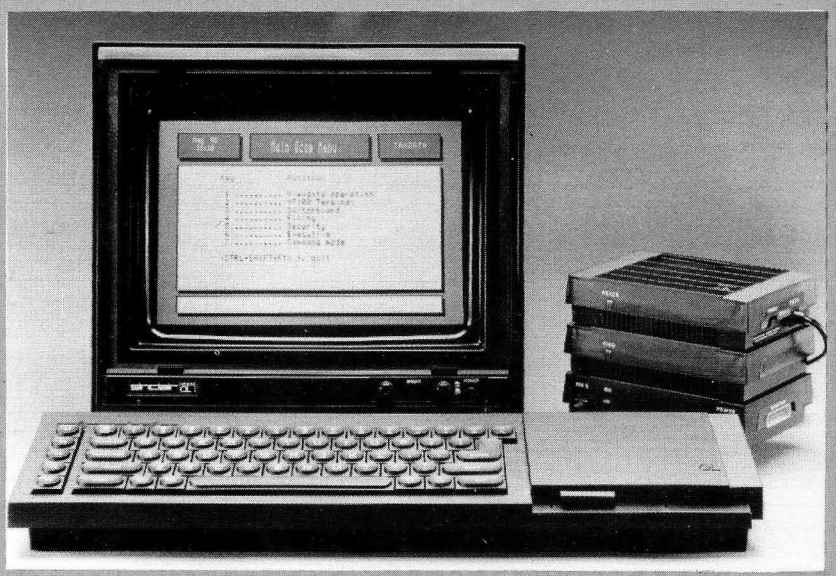
Address _____

_____ QLB006

Send to Sinclair Research Ltd., FREEPOST, Camberley, Surrey GU15 3BR

HARDWARE UPDATE

Things have been bullish on the QL hardware front this spring with disk drives seeming to sprout out of every corner and modems lurking among the ads in every new QL magazine.



Probably the biggest news is the announcement of an official QL printer, to be sold this summer.

The printer will provide output for graphics as well as text and offers both draft and near-letter-quality printing modes.

It is manufactured for Sinclair by a well-known printer company, and is designed to 'hook-up-and-go' with the QL. It is moulded, of course, in QL black and will fit in nicely with existing QL monitor.

Meanwhile, the communications options for the QL have taken a competitive turn – with two companies both announcing and delivering modem systems. The first is the QCOM communications system, which has been acquired by established modem manufacturers Tandata Marketing – following the collapse of OE Limited, which had originally developed the hardware component of the system.

Tandata also acquired the rights to communications software developed by Scicon for the QCOM system. Sinclair Research software manager Alison Maguire was integrally involved in the modem 'rescue' operation and says the modem is very important for the QL.

"The QCOM range of communications products has always been viewed as a vital element in opening up the QL's full, professional use. We are delighted with the commitment Tandata has shown in realising the QCOM range's commercial potential."

Although it is under no legal obligation to do so, Tandata is contacting all OEL customers who had placed orders, with a view to offering them a special package.

The Tandata modem will be competing with a new modem system from Modem House – who currently own the rights to market the famous Spectrum VTX5000 Prestel modem. Modem House's 'Bright Star' will feature Prestel access, terminal emulation and a parallel printer port.

STOP PRESS! It's just been announced that Sinclair Research will be offering own-label disk drive and interface units for the QL. Details are sketchy so far, but the plan is to offer two 3.5 inch 720K drives (with a total storage capacity of 1.4 Megabytes) and a disk interface for less than £500.

The system is designed and manufactured for Sinclair by MicroPeripherals and allows full integration with Qdos. More details on this as they become available.

"The QCOM range of communications products has always been viewed as a vital element in opening up the QL's full, professional use. We are delighted with the commitment Tandata has shown in realising the QCOM range's commercial potential." - Sinclair Research software manager Alison Maguire

PSION PROBLEM BOX

Problem: How to design QL Archive data files with large numbers of fields.

Answer: DISPLAY only prints the fields of a data file that it can fit on the work area. INSERT and ALTER only work with the fields visible on the work area. To use a data file with more fields than DISPLAY will print, you will need to design a screen or screens with SEDIT. A small procedure to swap between screens is a useful timesaver.

```
LET KEYS$="Z"
WHILE KEYS$()"q"
    PRINT AT 14,0;"PRESS A/B TO
    SWAP SCREEN [Q TO QUIT]"
    LET KEYS$=LOWER
    (GETKEY())
    IF KEY$="A":SLOAD "SCREE-
    NA":ENDIF
    IF KEY$ ="B":SLOAD
    "SCREENB":ENDIF
    ENDWHILE
```

As INSERT and ALTER work only on visible fields, it is useful to group the fields in the data file that are most frequently changed on one screen. If "SCREENB" contained the fields to be changed, selecting that screen and typing ALTER would allow the visible fields to be changed. To include the ALTER command in the above procedure add the line:

```
IF KEY$="C":SLOAD "SCREEN-
B":ALTER:ENDIF
```

into the WHILE loop. Another good way to pick out particular fields which you want to change is to use the SINPUT and UPDATE. If there is a field in the data file called name$ – and name$ has been included in the screen design – then SINPUT name$ will move the cursor to the location of that field in the screen design and allow that field to be changed. UPDATE writes the changed fields to the data file.

Programmers' forum

If you are a new SuperBASIC programmer, you may have asked yourself what an array is, and what can it be used for. If you wanted to be terribly formal, you could say that an array is a 'set of inter-related variables each of the same data type; each member of the set being accessed by the same identifier, subscripted by an index'. The problem with this is that it's almost meaningless to anyone but a computer scientist.

Let's put it in simpler terms. An array is a set of variables which are best considered as parts of a whole. Each of the variables in the set has the same 'type', which means that each variable in an integer array is an integer, each variable in a string array is a string, and so on. When we declare an array, we give it a single name (the 'identifier'). To be able to manipulate all the different variables inside the array, we must follow this identifier with a number (or numbers) in parenthesis. This number is known as the subscript.

a(3) = 23

This sets element number 3 to 23. Remember that the first element of an array is subscript 0. If we use a subscript to access an element which doesn't exist,

a(13) = 56

the program will stop with an error.

To get the value of an array element, you use exactly the same syntax, but on the right-hand side of an expression rather than the left:

print a$(1)

You can be really clever with arrays by giving them more than one dimension. The dimension is the number specified in the DIM statement which declares the array. If you put two numbers in the declaration,

100 DIM b(4,8)

then you create a two-dimensional array called 'b'. Probably the best way of looking at this is to consider it as a set of arrays. Our declaration above creates what is effectively five arrays of nine floating point variables, all of which are accessed using 'b' and two subscripts:

b(0,0)

accesses the first element of the array represented by b(0). You can go much further than two dimensions, but each added dimension multiplies the number of bytes of memory required by the array.

String arrays are a special case. They are declared in the same way, using the '$' suffix to specify string variables:

100 DIM b$(20),c$(3,15)

This declares a string array called 'b$' and a two-dimensional string array called 'c$'. The last dimension specified in a string array declaration (or the only one in the case of a one-dimensional array) is used to specify the length of each string in the array. This means that, from our declaration above, b$ is a string variable of fixed length (20 characters), and c$ is an array of four string variables, each of which is 15 characters long. Element 0 of the last dimension in a string array does exist, but should not be accessed as it is used to hold the string length.

Take an example: suppose this declaration was found in a program:

100 DIM a(5)

This line declares an array of six floating-point variables identified as a whole by 'a'. Two questions immediately come to mind. Why six variables, and why are they floating point? The DIM statement, which declares the array and reserves space inside the machine for its contents, creates enough space to cater for all the elements of 'a' from 0 to 5. This is different from the ZX Spectrum, which would have created an array of only five elements.

The variables inside the array are floating point variables for the same reason that a normal variable called 'a' would be. Unless specifically told otherwise, the SuperBASIC interpreter will make all identifiers floating point. This isn't important at the moment.

Now that the array 'a' exists, it is easy to put values into each element of the array, and just as easy to find out what those values are. We put a value into a particular element by using the number of the element as the array subscript:

Uses of arrays

Arrays are particularly useful whenever you want to store and manipulate numerous items which are related. As a typical, if unexciting example, consider an employee database management system. Each employee has a name, an address, a pay number and a salary. It seems eminently sensible to keep all this information in a group of arrays, so that a particular employee's record can be accessed by using the same subscript on each individual array.

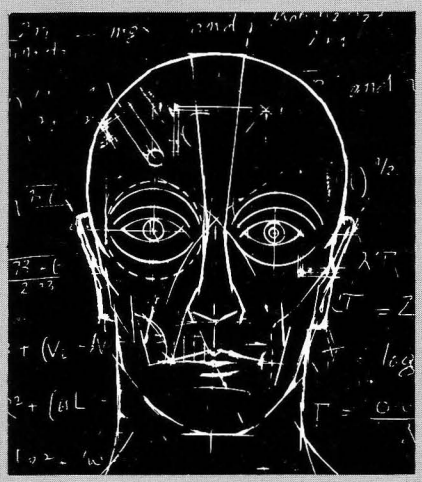
'Unless specifically told otherwise, the SuperBASIC interpreter will make all identifiers floating point.'

Qorporate QLose-up: Meeting Metacomco

Metacomco, the Bristol-based system software house which recently developed the C and Pascal programming languages (among others) for the QL, has a long history of participation in the personal computer market.

The company's first major software project in the micro field was to work with Digital Research (inventors of the well-known CP/M business operating system) in developing that company's Personal Basic. The original Basic source was sold by Metacomco to DR and then emerged as Personal Basic.

That was a few years ago and since then Metacomco has built a range of languages



and development products – it now includes Pascal, Basic, Lisp, Fortran, BCPL, editors, linkers, a multi-tasking operating system and a language dedicated to artificial intelligence.

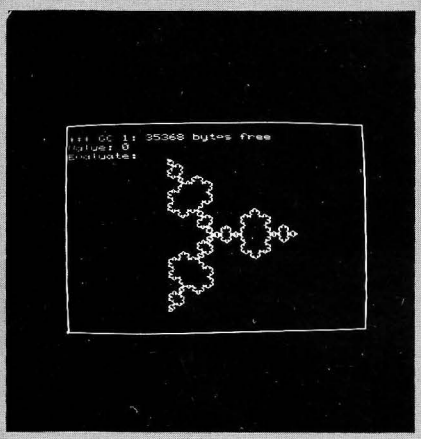
Not all the company's work is in system software. It has one interesting sideline aimed at reducing work involving algebraic equations for astronomers, mathematicians.

Until the QL products, most of Metacomco's work was on an OEM (Original Equipment Manufacturer) basis and they weren't known in the home computer market. It specialises – apart from Basic – in software for the 68000 processor family.

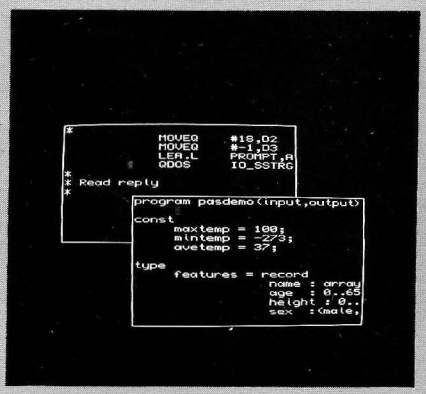
Metacomco took up development for the 68000 about 18 months ago.

The QL Assembler was launched first – and Metacomco says there was a strong demand for it – then BCPL, LISP and then Pascal.

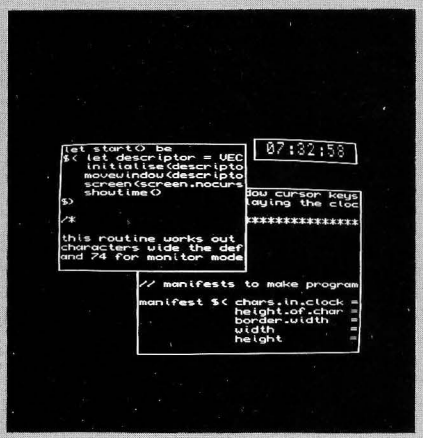
Aside from QL product development and releases, Metacomco still maintains a relationship with Digital Research and still promotes the Metacomco TRIPOS multi-tasking operating system – on which it is working to improve the user interface.



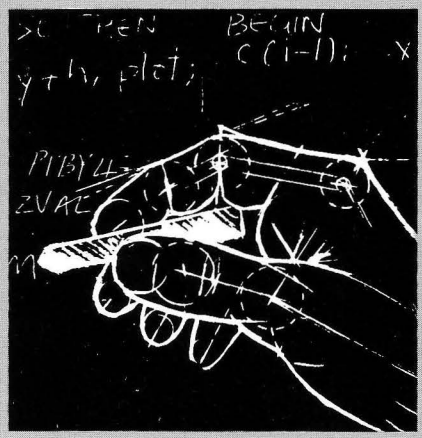
Lisp graphics: SNOWFLAKE

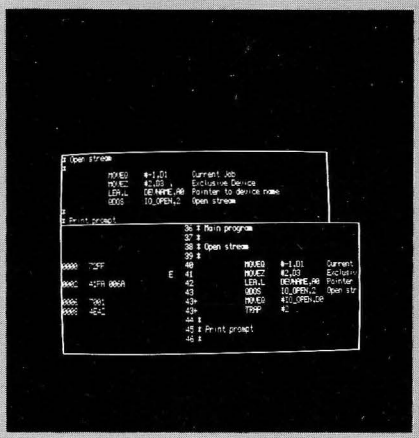


An assembler source code file and a Pascal source code file being edited simultaneously.



Two BCPL source code files being edited concurrently with the clock running as a third task.





An assembler source code file and the corresponding output listing from the assembly being edited concurrently.

OFF THE SHELF

To help QL owners get an idea of the books currently available we've compiled this list.

★ Inside the Sinclair QL, by Jeff Naylor and Diane Rogers, £6.95. A guide to the inner workings of the QL, with some useful illustrated programs.

★ Artifical Intelligence on the Sinclair QL, by Keith and Steven Brain, £6.95. This book helps your QL get to grips with Artificial Intelligence and make it an orator, a teacher, even a pupil. All this including AI routines.

★ Assembly Language Programming on the Sinclair QL, by Andrew Pennell, £7.95. An introduction to machine code explaining how the QL uses its 68000 family processor and includes a collection of proven progam routines for immediate use.

★ Essential Mathematics on the Sinclair QL, by Czes Kosniowski, £6.95. Provides basic programming routines including an explanation of all the mathematical routines and illustrations in short programs. Packed with maths information.

★ Developing Applications for the Sinclair QL, by Malcolm Davison, £7.95. Using examples drawn from years of business and home applications, this guide unleashes a host of powerful QL software business and home applications.

★ Introduction to Simulation Techniques on the Sinclair QL, by John Cochrane, £6.95. An introduction to the Sinclair QL's Simulation Techniques.

★ Practical Software for the Sinclair QL, by David Lawrence, £6.95. This solid introduction to useful programming is a library of working routines, exploring techniques and detailing an entire collection of applications programs using a modular format.

★ The Sinclair QL Companion, by Boris Allen, £6.95.

★ Exploring the Sinclair QL, by Andrew Nelson, £4.95.

★ Mathematics on the Sinclair QL. Examines the use of mathematical signs including COS ,ABS ,AND, SGN, in writing programs with specific reference to the QL. These mathematical utilities are illustrated in short programs which can be integrated into larger ones.

★ The Working Sinclair QL, by David Lawrence. A more general book explaining computer applications in areas such as finance, tax, information storage, graphics, music and education.

★ QL Superbasic, by A. A. Berk, £5.95. Following the usual fare for teaching the QL's special dialect of Basic/SuperBASIC, this book offers an additional chapter on Microdrive storage and handling, plus an appendix on system commands.

★ QL Computing, by Ian Sinclair, £5.95. Examines the facilities the machine offers and provides programs demonstrating them covering graphics, sound, database applications and fault finding.

★ Sinclair QL Adventures, by Tony Bridge and Richard Williams, £5.95. Write your own adventure with this book using its specially devised Generator program.

★ Advanced Graphics Routines for Your Sinclair QL, by Alan Rudge, £6.95. Learn how to take full advantage of the QL's graphics facilities with this useful guide.

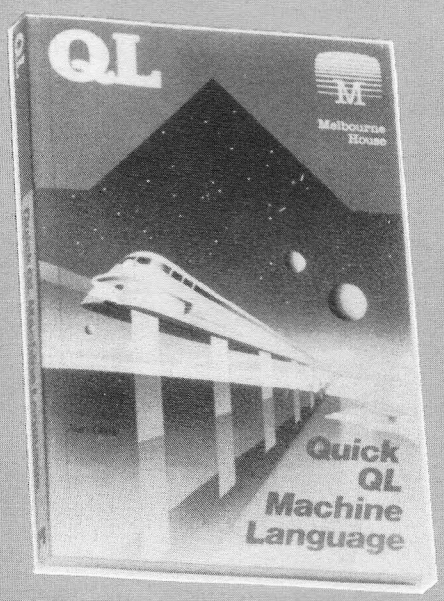
★ Basic Programming on the QL, by Neil Cryer and Pat Cryer, £7.95. Beginning with fundamentals, this book provides a comprehensive, illustrated course on QL SuperBASIC. After mastering this you can move on to more complex tasks and test everything learned with the suggested activities and sample programs.

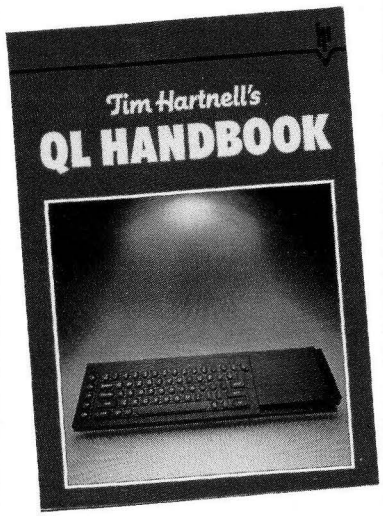
★ The QL Book of Games, by Richard G. Hurley and David D. Virgo, £6.95. You can build your own games library and key-in the programs while learning how to operate the SuperBASIC with this guide.

★ QL Superbasic: A Programmer's Guide, by John Wilson, £6.95. Chapters cover the SuperBASIC's features focusing on Structure, Procedures and Functions, Condition Testing, Repetition Control, Arrays, Strings, the QL Display, Sound, Timing, Filing and User Friendliness.

★ The Sinclair QL Series. This series is a collection of five books designed to help you get the most from your QL:

Introducing the Sinclair QL..... £6.95
Introduction to SuperBASIC on the Sinclair QL..... £6.95
Advanced Programming with the Sinclair QL..... £6.95
Desk Top Programming with the Sinclair QL..... £6.95
Word Processing with the Sinclair QL £6.95





★ QL Abacus, by Spottiswoode. A guide to using Abacus spreadsheets.

★ Quick QL Machine Language, by Giles. This book teaches the basics of how to program your QL in machine language.

★ QL Quill, by Simon and Spottiswoode. A how-to guide for using the QL Word Processor.

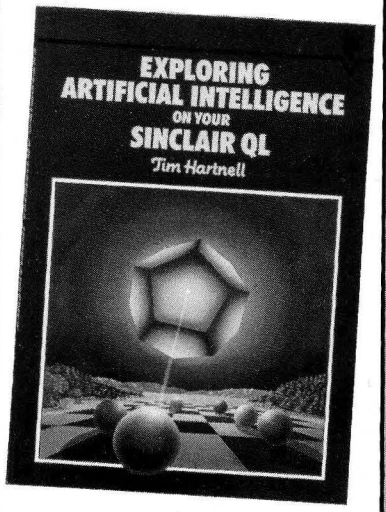
★ QL Easel, by Spottiswoode. A how-to for QL Graphics.

★ QL SuperBASIC by A.A. Berk. For the neophyte QL SuperBASIC user.

★ QL Gamesmaster, by Kay Ewbank, Mike James and S.M. Gee, £7.95. A well-presented compromise between the basic programmers' guide and a book of advanced programs.

★ Getting to Know Your Sinclair QL, by Stephen Morris, £7.95. For the nervous or novice user, pre-manual guide to the QL providing real nuts and bolts stuff.

★ Exploring Artificial Intelligence on Your Sinclair QL, by Tim Hartnell, £6.95. A highly readable and painless demonstration of the AI techniques discussed. Also includes a discussion of learning and reasoning concepts, and a look at Expert systems, the fast-growing offshoot of AI.



★ QL Advanced User Guide, by Adrian Dickens, £12.95. A complete guide to Qdos and the Sinclair QL, it covers multi-tasking, transient programs, resident procedures, heaps and stacks, traps and utilities, and 68000 assembler programming. All programs are also available on Microdrive cartridge.

★ Quantum Theory: A Guide to the Sinclair QL, by Jeremy San, Fouad Katan and Simon Rockman, £5.95. Provides a down-to-earth practical guide to learning SuperBASIC and QL programming.

★ QL SuperBASIC: The Definitive Handbook, by Jan Jones. Written by the designer and writer of Sinclair's QL SuperBASIC language, this book is far more than a detailed description of commands. It provides an explanation of the language's structure and its design. For the beginner it demonstrates the elegance, speed and efficiency of the QL's power.

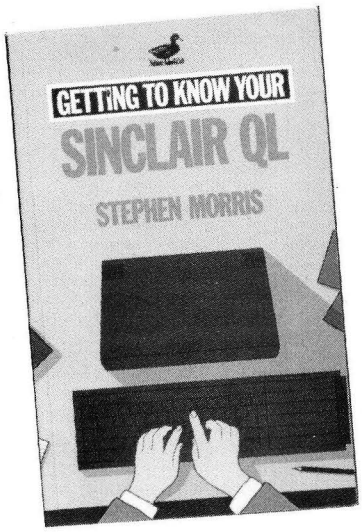
★ QL SuperBASIC: A Programmer's Guide, by John Wilson, £6.95. Gives programmers a good understanding of the differences between the SuperBASIC and BASIC.

★ QL Assembly Language Programming by Colin Opie, £6.95. Intended for use with McGraw Hill's program editor/68000 assembler package costing £29.95, this is an advanced, technical guide to 68000 assembly language.

★ QL Games Compendium, by Tim Hartnell, £6.95. Twenty-three games listed under Adventure, Artificial Intelligence, Board Games, Deduction and Perception.

★ A QL Compendium, by Gandoff and King. This is a collection of QL program listings published by Addison-Wesley.

'Getting to Know Your Sinclair QL . . , is a book for the nervous or novice user'



★ QL Handbook, by Tim Hartnell. This is a general guide to the QL computer.

★ Professional and Business Uses of QL, by Lewis. A basic guide for the QL business user.

★ The C Programming Language, by Brian Kerninghan and Dennis Ritchie, £22.95, Written by the man who created C and implemented it on the Unix operating system on Digital Equipment Corporation's PDP-11 computers, this is a guide for keen programmers casting around for information on the subject.

★ Learning to Program in C, by Thomas Plum, £15.95. Not intended for home users, this guide is designed to turn the reader into a competent programmer in a real software environment.

★ Logic, Algebra and Databases, by Peter Gray, £9.95. Examines the applications of logic programming and applications programming to database through the database query languages; this guide is aimed at students, postgraduates and programmers interested in the topic.

★ 68000 Assembly Language Programming, by Kane, Hawkins, Levanthal, £19.50.

Machine Code

All the SuperBASIC graphics procedures, such as LINE, CIRCLE and POINT, make the use of graphics on the QL extremely simple. If you decide to delve into machine code, you'll find that all the graphics routines have their origins in assembly language, as each procedure is available, near enough, by calling various parts of Qdos into play.

When you first look at the descriptions of the Qdos graphics routines in the QL Technical Guide, things look depressingly difficult – floating point co-ordinates, RI stacks and channel IDs.

It's nothing like so difficult as it seems, though. Once you understand the concepts involved, you'll find yourself writing graphics routines in machine code almost as easily as you would write them in SuperBASIC.

The graphics routines are all part of the 'redirectable I/O system', which is accessed via the 68000 TRAP #3 instruction. Before executing this instruction you must put a value into register D0 which tells the system exactly what action to take. The codes we are interested in here are shown below:

Value	Routine Name	Action
$30	SD.POINT	Plots a point
$31	SD.LINE	Draws a line
$32	SD.ARC	Draws an arc
$33	SD.ELPIS	Draws an ellipse
$34	SD.SCALE	Sets graphics scale
$36	SD.GCUR	Sets graphics cursor

(The '$' means that the numbers are in hexadecimal)

Apart from having one of these values in D0, the routines also require three other pieces of information. The first is held in register D3, and is known as the timeout. This tells the routine how long it must wait before it returns with failure. If the process carried out by the routine cannot be completed in this time, the routine returns with error 'not complete'. The time is measured in units of 20ms, so a timeout of 50 corresponds to 1 second. We can put a special value in D3, -1, which means 'infinite timeout' and causes the routine to return only when it has completed its task.

Listing 1

```
* A (very) simple graphics demo by Adam Denning

          SIZE      300

STACK     EQU       -50

IO.OPEN   EQU       1
RI.EXEC   EQU       $11C
RI_FLOAT  EQU       8
SD.LINE   EQU       $31
MT.FRJOB  EQU       5

          BRA.S     START
          DC.L      0
          DC.W      $4AFB
          DC.W      8
          DC.B      'GRAPHIC1'

START     LEA.L     SCR_NAME,A0
          MOVEQ     #0,D3
          MOVEQ     #-1,D1
          MOVEQ     #IO.OPEN,D0
          TRAP      #2

          LEA.L     CO_ORDS,A2
          LEA.L     STACK(A5),A1        Make A1 into A6-relative RI stack pointer

          MOVEQ     #3,D1               We're going to convert 4 ints to fps

CONVERT   SUBQ.L    #2,A1               make room for integer
          MOVE.W    (A2)+,0(A6,A1.L)
          MOVE.W    RI.EXEC,A3          Convert to floating point - let QDOS
          MOVEQ     #RI_FLOAT,D0        do all the hard work!
          JSR       (A3)
          DBRA      D1,CONVERT          Repeat for all four integers

          TRAP      #4                  Tell IOSS the addresses are A6-relative

          MOVEQ     #SD.LINE,D0         Draw the line
          MOVEQ     #-1,D3              ...take your time
          TRAP      #3

          MOVEQ     #0,D3               out, out damn spot!
          MOVEQ     #-1,D1
          MOVEQ     #MT.FRJOB,D0
          TRAP      #1

SCR_NAME  DC.W      4
          DC.B      'SCR_'

CO_ORDS   DC.W      10,20               X,Y co-ordinates of line start
          DC.W      30,50               X,Y co-ordinates of line end

          END
```

and more....

The second bit of information is the channel ID. This is the internal number used by Qdos to identify uniquely each open channel. It has nothing at all to do with the '#' numbers used by the SuperBASIC interpreter to identify its channels. This number is returned by the Qdos routine which opens channels, assuming it succeeded, and is almost always required in register A0. The graphics routines are no exception.

Finally, each of the routines needs a pointer to its arguments. These arguments are held on a special type of stack, called the 'Arithmetic' or 'RI' stack. As stacks go, there is nothing unusual about it except that register A1 is used as the stack pointer and, as things can move about in memory from time to time, the value held in A1 is relative to A6. This means that although A1 can be considered as the RI stack pointer, the actual value of an argument will be found only by using A1 as an index to A6:

Real_SP = 0(A6,A1.L)

so

top_of_stack_value = 0(A6,A1.L)

It is not this A6 relative value of the RI stack pointer which needs to be passed to the graphics routines, though. These need the RI stack pointer in register A1 to be absolute. We'll see later that we can alleviate the problem of some routines needing A6 relative and others not by taking advantage of another Qdos routine invoked by TRAP #4.

Sounds trivial so far, doesn't it? Things are complicated a little (well OK, a lot) by the fact that the arguments to each of the graphics routines must be in internal floating point format.

There's no easy way of calculating the internal representation of a given number, but once again Qdos can come to our aid. So long as the numbers we want to convert are whole numbers between -32768 and 32767 (i.e. 16-bit integers), we can use a Qdos routine called RI.EXEC to convert the number into floating point format for us.

RI.EXEC is not invoked by a 68000 TRAP instruction, as it is a vectored routine. This means that the symbol RI.EXEC equates to the address of a vector. Inside this vector is the address of the routine itself. The normal way of calling a vectored routine is to put the address of the vector into an address register and then execute a JSR (An) instruction.

RI.EXEC performs a number of operations, most of them on floating point numbers, but it can also do a few conversions. The code representing the operation is put into register D0 before the call and the RI stack pointer (A6 relative) must be in register A1. After the call, the RI stack pointer is updated to point to the result of the operation, D0 holds the error code (0 means success), and all the other registers are preserved.

Listing 2

```
* Another (very) simple graphics demo by Adam Denning
* This time using 'random' numbers to draw lines all over the place
         SIZE     300

STACK    EQU      -50

IO.OPEN  EQU      1
RI.EXEC  EQU      $11C
RI_FLOAT EQU      8
SD.LINE  EQU      $31
SD.SETIN EQU      $29

         BRA.S    START
         DC.L     0
         DC.W     $4AFB
         DC.W     8
         DC.B     'GRAPHIC2'

START    LEA.L    SCR_NAME,A0
         MOVEQ    #0,D3
         MOVEQ    #-1,D1
         MOVEQ    #IO.OPEN,D0
         TRAP     #2

AGAIN    LEA.L    STACK(A5),A1      Make A1 into A6-relative RI stack pointer
         MOVEQ    #3,D1             We're going to convert 4 ints to fps

CONVERT  BSR.S    RANDOM            get random number
         SUBQ.L   #2,A1             make room for integer
         ANDI.W   #$7F,D4           less random than before?
         MOVE.W   D4,0(A6,A1.L)     stack random number
         MOVE.W   RI.EXEC,A2        Convert to floating point - let QDOS
         MOVEQ    #RI_FLOAT,D0      do all the hard work!
         JSR      (A2)
         DBRA     D1,CONVERT        Repeat for all four integers

         TRAP     #4                Tell IOSS I love her...

         MOVEQ    #SD.LINE,D0       Draw the line
         MOVEQ    #-1,D3            ...take your time
         TRAP     #3                the bastard corrupts A1

         BSR.S    RANDOM            change the ink to a random colour
         MOVE.W   D4,D1
         MOVEQ    #SD.SETIN,D0
         TRAP     #3

         BRA.S    AGAIN             and do it all over again

RANDOM   MOVE.L   (A4)+,D4          hopeless 'algorithm', but it'll do
         ROL.L    #3,D4
         MOVE.L   (A3)+,D5
         EOR.L    D5,D4
         ROR.L    #5,D4
         RTS

SCR_NAME DC.W     4
         DC.B     'SCR_'
         END
```

Machine Code

The RI.EXEC operation we are interested in is called RI_FLOAT, which converts a 16-bit signed integer on the RI stack to the corresponding floating point number. An integer takes up two bytes, while a floating point number takes up six, so the RI stack pointer (A1) will be four less than it was prior to the call.

Let's put all this theory to work now by writing a very simple program to draw a line somewhere on the screen. Such a program would use the SD.LINE routine, which needs the following parameters on the RI stack:

$00(A1) Y co-ordinate of line end
$06(A1) X co-ordinate of line end
$0C(A1) Y co-ordinate of line start
$12(A1) X co-ordinate of line start

As the Y co-ordinate of the end of the line is the last thing on the stack (think about it), we can stack things in the logical order of (X,Y) start of line, (X,Y) end of line.

Listing 1 is one way of implementing our example program. Notice that we set the program's data space size to 300 bytes here, as the Qdos documentation says that at least 240 bytes must be available on our RI stack whenever one of these routines is called.

The first thing the program does is to open a screen channel for itself so that we can actually see the line it's going to draw. The open is done by loading register A0 with the address of the name of the device to be opened ('SCR_'), putting -1 into D1 (to signify 'open for this job') and 0 into D3. D3 is used to indicate the type of channel open required (read only, write only, overwrite, etc.). Finally, the value of IO.OPEN (1) must be put into register D0 and a TRAP #2 instruction executed. The routine returns with the channel ID allocated in A0 if there were no errors. Our program assumes that there have been no errors.

The rest of the program draws the line. First, each of the four co-ordinates must be converted into floating point numbers on the RI stack. We create an ad hoc RI stack in the job's data area, using a negative offset from A5. When a job is first started, register A6 is set to point to the beginning of the job, and 0(A5,A6.L) points to the end of the job's data space. A5 is therefore relative to A6, so -50(A5) is also relative to A6. A1 is now our RI stack pointer.

Each integer co-ordinate occupies two bytes on the RI stack, so 2 must be subtracted from A1 each time an integer is stacked. The thus-stacked integer is then converted to internal floating point format by RI.EXEC. D1 controls a DBRA loop to convert and stack all four co-ordinates. Once the loop has exited, the RI stack is almost ready for SD.LINE. The only problem is that, as was said earlier, the graphics routines need A1 to be absolute rather than A6 relative. This could be solved by adding A6 to A1 with ADDA.L A6,A1, but we may as well take advantage of another Qdos routine. Whenever TRAP #4 is executed, it makes the next I/O routine (anything invoked by TRAP #3) consider any addresses passed to it to be relative to A6.

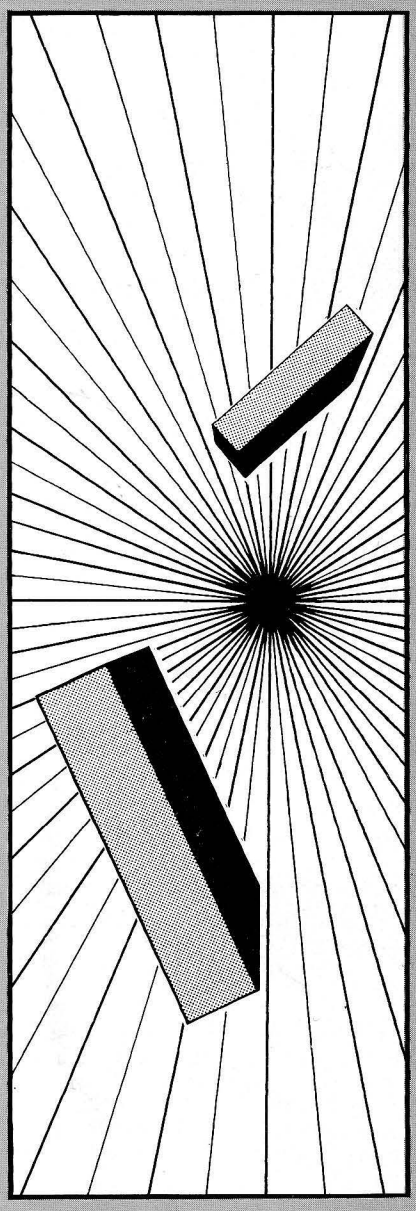
Once the line has been drawn, the job is killed with the MT.FRJOB manager trap. When the program has been assembled, it can be run with either EXEC or EXEC W.

The other program here, shown in listing 2, is a bit more ambitious. It continuously draws random lines in random colours.

Much of the code is the same as the previous sample but each co-ordinate is calculatd randomly. The subroutine which does this RANDOM is by no means a true random number generator. In fact, it only plays around with a couple of numbers and would not normally be useful for any purpose requiring 'more random' numbers. Nevertheless, it suffices here. Four random numbers between 0 and $7F are stacked at a time, and then SD.LINE is called to draw the line. After each line has been drawn, RANDOM is called again to get a new value for the screen's ink colour, and the TRAP #3 routine SD.SETIN is called to perform the colour change. The program then loops to do it all over again, hopefully with different random numbers.

With the QL Technical Guide in one hand and a QL Assembler in the other, you can draw circles, arcs, blocks, borders and so on with very little effort.

With the QL Technical Guide in one hand and a QL Assembler in the other, you can draw circles, arcs, blocks, borders and so on with very little effort.



QL Quarks

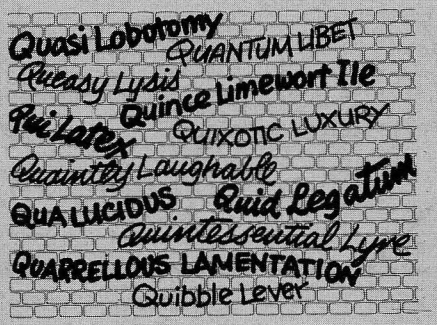
The most difficult peripheral. The reviewing skills of the QLUB staff were recently over-taxed with the arrival of a device which didn't plug into the QL's expansion port, used no electricity, had no accompanying software - but was still decorated in beautiful QL black. It was a dustcover.

Actually, it was dustcover number one to pass through our office doors. In our first benchtest, this particular dustcover failed our primary criterion – does it fit the QL? We were left scratching our heads – either our QLs have been using steroids or they must have a pretty funny ruler back at the factory.

Dustcover number two which promptly arrived to replace dustcover number one – sent back until it grew enough – proved to be a perfect fit.

Unfortunately, this did not solve the problem of the review - what remotely interesting things can you say about a dustcover? It probably has about as much literary appeal as a dissertation on a shoehorn.

Well, we can tell you it's black and it keeps out the dust while it covers the QL. The only other thing we can add is that it was the first of a number of QL dustcovers we were to see. They range in price from £3.50 to £5.95 – but try them before you buy.



A smart book buy?

We recently came across the smartest-looking book for the QL – this one should easily qualify for entrance into the MENSA QL Users' Library. The title of this airy tome: Artificial Intelligence on the QL by Keith and Steven Brain.

QL on the range

WHAT THE PAPERS SAY: A rather interesting advert popped up recently in a popular weekly computing magazine which shall remain nameless:

FOR SALE: Sinclair QL with ungrazed quilt, £300 or offers?

QL mast storage

For those of you who would like something a little different for this year's holiday – why not try a Weymouth company's new packaged extravaganza – a computing windsurfing holiday? Don't ask how or why, but we'd like to hear from any survivors.

SPECIAL OFFER OFFICE

The Special Offer Office keeps track of all deals and discounts for QLUB members – which includes anybody who is sent this newsletter. Listed below are the deals offered to members;

★ Metacomco is planning a special limited offer for QLUB members on its development and programming software. For more details, write before August 15 to:

Metacomco offer
Special Offer office
26 Portland Square
Bristol, BS2 8RZ.

★ GST Computer Systems is offering its acclaimed 68K/OS operating system for the QL at an attractive discount. For details, write to GST Computer Systems, 8 Green St., Willingham, Cambridge CB4 5JA or telephone Jane Pateman on (0954) 81991. Have your membership number and address to hand.

★ A 20 percent discount is available on selected items of QL software sold by Sinclair Research through QLUB.
QL Monitor – QLUB price £19.95.
QL Assembler – QLUB price £31.95
QL Toolkit – QLUB price £19.95

Send your order using the form on page 3 to Sinclair Research, Stanhope Road, Camberley, Surrey or telephone on (0276) 685311.

★ **A discount is available for a year's subscription to EMAP's QL User magazine.** A full year's subscription including delivery would normally cost £15. If you subscribe through QLUB, the magazine will be delivered for an all-in (postage included) price of £11.50. To receive your 12 issues, send a cheque or money order for £11.50 to PRQL, Subscription Department, QL User, Priory Court, 30/32 Farringdon Lane, London EC1.

★ Westway software is offering its new machine-code arcade game - EVA – at a discount to QLUB members. The game normally sells for £10.95 – but QLUB members can buy it for £1 off that price. Send your orders to Westway, 24 Preston Road, Lytham, Lancashire.

NEXT ISSUE

- More machine code
- Printer details
- Disk system examination
- Psion problem page